

REMARKS

Reconsideration of the application in view of the following remarks is respectfully requested. No claims have been amended. No claims have been canceled. New Claims 70-78 have been added. Claims 1-78 are currently pending in the application.

CLAIMS 2, 25, AND 45

Paragraph 2 of the Office Action rejected Claims 2, 25, and 45 under 35 U.S.C. §112, first paragraph, as containing subject matter which was not described in the specification in such a way as to enable one skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the invention. Specifically, the Office Action questioned where the specification disclosed the claim feature: “said wrapper instance is invocable by the application without further interaction with the framework.”

In response to this rejection, Applicant submits that support for this feature is found in the specification at page 3, lines 19-22, which read: “Since the customized implementation incorporates the restrictions and enforcement logic for enforcing the restrictions, it is not necessary for the application to further interact with the framework.” Applicant further submits that support for the above feature is found in the specification at page 7, lines 21-23, which read similarly to the above quoted language. Applicant submits that the subject matter contained in Claims 2, 25, and 45 is described in the specification in such a way as to satisfy the requirements of 35 U.S.C. §112, first paragraph. Therefore, Applicant respectfully requests that this rejection be withdrawn.

Applicants note that Claims 2, 25, and 45 were rejected only under 35 U.S.C. §112. They were not substantively rejected under 35 U.S.C. §102 or §103. As argued

above, Claims 2, 25, and 45 satisfy the requirements of 35 U.S.C. §112. Therefore, Applicants submit that Claims 2, 25, and 45 are allowable.

CLAIMS 1, 24, AND 47

Paragraph 3 of the Office Action rejected Claims 1, 3-24, 26-47, and 49-69 under 35 U.S.C. §102(e) as being anticipated by Elgamal et al. (U.S. Patent No. 6,389,534 B1). This rejection is respectfully traversed.

With regard to Claim 1, there is recited a method performed by a framework in a system comprising the framework, an application, and an implementation class that provides an implementation for a particular service; that method comprising:

receiving a request from an application for a customized **implementation** of a particular service . . .
instantiating a **wrapper class** to give rise to a **wrapper instance**, said wrapper instance comprising enforcement logic for enforcing said restrictions;
encapsulating said implementation instance and said restrictions within said wrapper instance; and
providing said **wrapper instance** to the application as said customized implementation.

(emphasis added).

The method of Claim 1 is quite advantageous because it allows an application to obtain access to services without repeatedly requesting those services from some centralized framework. Specifically, when an application needs to access a particular service, it makes a request to a centralized framework. The request is a request for an **implementation** of the particular service, rather than a request for a result from the particular service.

The centralized framework instantiates a wrapper class to give rise to a wrapper instance. The instance is a **wrapper instance** because it is designed to

encapsulate another instance. The wrapper instance includes enforcement logic for enforcing certain restrictions on the requested implementation. Thus, the implementation itself need not be modified.

An instance of the requested implementation is **encapsulated** within the wrapper instance. Thus, access to the implementation instance can only be had through the wrapper instance. This prevents the requesting application from circumventing the restrictions.

The **wrapper instance** that encapsulates the implementation instance is then provided to the requesting application as the requested implementation. It is a re-usable implementation, rather than a single result of an operation, that is returned to the application. Thus, after making an initial request to the centralized framework for an implementation of a service, the application does not need to repetitively request that service from the centralized framework again. This removes the centralized framework as a performance bottleneck. These and other benefits can be derived from the method of Claim 1.

ELGAMAL

Elgamal does not disclose such a method. Instead, Elgamal discloses an application program calling a service module, and the service module, in turn, calling a policy filter that has been configured by a policy filter initialization module. Based on the determination of its policy filter, the service module returns to the application either an error (if the operation is not allowed) or a result of the operation requested by the application program (see col. 5, line 41-42, and also col. 6, lines 42-43).

An important point to note regarding Elgamal is that, unlike the method of Claim 1, the application of Elgamal requests an operation, NOT an implementation of a service. Thus, every time that the application of Elgamal requests an operation, it must request the operation from some centralized framework (e.g., the APIs, protocols, and services, and policy filters shown in FIG. 1 of Elgamal). This can create a performance bottleneck at the centralized framework; a disadvantage avoided by the method of Claim 1, as described above.

Another important point to note regarding Elgamal is that, unlike the method of Claim 1, a service module is configured by a policy filter initialization module; a wrapper instance is NOT instantiated from a wrapper class. Even if each service module is considered to be a class, Elgamal fails to disclose that other instances of these service modules are ever instantiated. While Elgamal discloses that these service modules may be configured, such configuration merely alters the service module so that the former configuration of that service module is not maintained. Therefore, configuration is not equivalent to instantiation.

Elgamal also fails to disclose a wrapper class. The service modules of Elgamal do not include their own instances of the cryptographic modules. Thus, the service modules of Elgamal cannot be considered to “wrap around” the cryptographic modules in the same manner that the wrapper instances of Claim 1 encapsulate implementation instances (that provide cryptographic functionality). Consequently, Elgamal fails to disclose wrapper classes or instances of those wrapper classes such as are used by the method of Claim 1.

Another important point to note regarding Elgamal is that, unlike the method of Claim 1, the service module calls the cryptographic module (see col. 6, lines 40-42); the service module does NOT encapsulate the cryptographic module. Because Elgamal contemplates only one instance of each kind of cryptographic module (FIG. 1 of Elgamal only shows one instance of each crypto plug-in), multiple service modules that call that cryptographic module must share that cryptographic module. This can create a performance bottleneck such as is avoided by the method of Claim 1, as described above.

Another important point to note regarding Elgamal is that, unlike the method of Claim 1, the service module of Elgamal returns to the application either an error or a result of an operation, NOT a wrapper instance. The service module is part of a centralized framework (only one instance of each service is shown in FIG. 1 of Elgamal). Each application of Elgamal must make a request of the same centralized framework whenever a result is desired. This can create a performance bottleneck at the centralized framework; a disadvantage avoided by the method of Claim 1, as described above. For at least these reasons, Applicant submits that Elgamal does not anticipate Claim 1.

Claim 24 is a device claim analogous to the method of Claim 1. Claim 47 is a computer-readable medium claim analogous to the method of Claim 1. Applicant submits that Claims 24 and 47 are not anticipated by Elgamal for at least the reasons given above in connection with claim 24.

REMAINING DEPENDENT CLAIMS

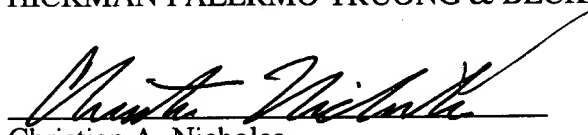
The pending claims not discussed so far are dependent claims that depend on an independent claim that is discussed above. Because each of the dependent claims include the limitations of claims upon which they depend, the dependent claims are patentable for at least those reasons the claims upon which the dependent claims depend are patentable. Removal of the rejections with respect to the dependent claims and allowance of the dependent claims is respectfully requested. In addition, the dependent claims introduce additional limitations that independently render them patentable. Due to the fundamental difference already identified, a separate discussion of those limitations is not included at this time.

For at least the reasons set forth above, Applicant respectfully submits that all pending claims are patentable over the art of record, including the art cited but not applied. Accordingly, allowance of all claims is hereby respectfully solicited.

Respectfully submitted,

HICKMAN PALERMO TRUONG & BECKER LLP

Dated: July 26, 2002


Christian A. Nicholes
Reg. No. 50,266

1600 Willow Street
San Jose, California 95125-5106
Telephone No.: (408) 414-1080
Facsimile No.: (408) 414-1076

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Box Amend, Commissioner for Patents, Washington, DC 20231.

on 7/26/02 by Trudy Bagdon
(Date) (Signature)



Marked Version of Claims

1 1. (Not Amended) In a system comprising an application, a framework, and
2 an implementation class which provides an implementation for a particular service, a
3 method performed by the framework, comprising:

4 receiving a request from an application for a customized implementation of a
5 particular service;

6 instantiating an implementation class which provides an implementation for the
7 particular service to give rise to an implementation instance;

8 determining a set of zero or more restrictions to be imposed on said customized
9 implementation;

10 instantiating a wrapper class to give rise to a wrapper instance, said wrapper
11 instance comprising enforcement logic for enforcing said restrictions;

12 encapsulating said implementation instance and said restrictions within said
13 wrapper instance; and

14 providing said wrapper instance to the application as said customized
15 implementation.

1 2. (Not Amended) The method of claim 1, wherein said wrapper instance is
2 invocable by the application without further interaction with the framework.

1 3. (Not Amended) The method of claim 1, wherein the implementation class
2 provides an unrestricted implementation for the particular service.

1 4. (Not Amended) The method of claim 3, wherein the particular service is
2 an encryption/decryption service, and wherein the unrestricted implementation provided
3 by the implementation class is capable of using unlimited encryption/decryption
4 parameters.

1 5. (Not Amended) The method of claim 4, wherein the unrestricted
2 implementation provided by the implementation class is capable of using
3 encryption/decryption keys of any size.

1 6. (Not Amended) The method of claim 1, wherein said enforcement logic
2 enforces said restrictions on said implementation instance.

1 7. (Not Amended) The method of claim 6, wherein said enforcement logic
2 enforces said restrictions on said implementation instance by:
3 receiving a set of desired parameters from the application;
4 determining whether the desired parameters exceed said restrictions; and
5 in response to a determination that the desired parameters exceed said restrictions,
6 preventing said implementation instance from operating.

1 8. (Not Amended) The method of claim 7, wherein said enforcement logic is
2 invoked upon initialization of said wrapper instance.

1 9. (Not Amended) The method of claim 1, wherein the system further
2 comprises an exemption mechanism class which provides an implementation for a
3 particular exemption mechanism, and wherein said method further comprises:
4 instantiating the exemption mechanism class to give rise to an exemption
5 mechanism instance; and
6 encapsulating said exemption mechanism instance within said wrapper instance.

1 10. (Not Amended) The method of claim 9, wherein said enforcement logic is
2 invoked upon initialization of said wrapper instance, and when invoked, said enforcement
3 logic:
4 determines whether said exemption mechanism instance has been invoked; and
5 in response to a determination that said exemption mechanism instance has not
6 been invoked, preventing said implementation instance from operating.

1 11. (Not Amended) The method of claim 1, wherein said wrapper instance
2 comprises one or more invocable methods, wherein said implementation instance
3 comprises one or more invocable methods, and wherein encapsulating comprises:
4 mapping the one or more invocable methods of said wrapper instance to the one
5 or more invocable methods of said implementation instance.

1 12. (Not Amended) The method of claim 1, wherein instantiating the
2 implementation class comprises:
3 determining whether the implementation class is authentic; and

4 in response to a determination that the implementation class is authentic,
5 instantiating the implementation class to give rise to said implementation instance.

1 13. (Not Amended) The method of claim 12, wherein the implementation
2 class has a digital signature associated therewith, and wherein determining whether the
3 implementation class is authentic comprises:
4 verifying said digital signature.

1 14. (Not Amended) The method of claim 12, wherein the implementation
2 class authenticates the framework prior to giving rise to said implementation instance.

1 15. (Not Amended) The method of claim 1, wherein determining the set of
2 zero or more restrictions comprises:
3 accessing information specifying one or more limitations; and
4 processing said limitations to derive said restrictions.

1 16. (Not Amended) The method of claim 15, wherein the particular service is
2 an encryption/decryption service, and wherein said information comprises a set of one or
3 more default encryption limitations.

1 17. (Not Amended) The method of claim 16, wherein said default encryption
2 limitations are derived by merging multiple jurisdiction policies and extracting therefrom
3 the most restrictive encryption limitations.

1 18. (Not Amended) The method of claim 1, wherein determining the set of
2 zero or more restrictions comprises:
3 accessing information specifying one or more limitations;
4 determining permissions, if any, granted to the application; and
5 reconciling said limitations and said permissions to derive said restrictions.

1 19. (Not Amended) The method of claim 18, wherein said limitations and said
2 permissions are reconciled to derive restrictions which are least restrictive.

1 20. (Not Amended) The method of claim 18, wherein the particular service is
2 an encryption/decryption service, and wherein said information comprises a set of one or
3 more default encryption limitations, and a set of zero or more exempt encryption
4 limitations which apply when one or more exemption mechanisms are implemented.

1 21. (Not Amended) The method of claim 20, wherein said default encryption
2 limitations and said exempt encryption limitations are derived by merging multiple
3 jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

1 22. (Not Amended) The method of claim 20, wherein reconciling said
2 limitations and said permissions comprises:
3 determining whether the application has been granted any permissions; and
4 in response to a determination that the application has not been granted any
5 permissions, deriving said restrictions from said set of default encryption limitations.

1 23. (Not Amended) The method of claim 20, wherein reconciling said
2 limitations and said permissions comprises:
3 determining whether the application has been granted any permissions which
4 require implementation of a particular exemption mechanism;
5 in response to a determination that the application has been granted a permission
6 which requires implementation of a particular exemption mechanism, determining
7 whether said exempt encryption limitations allow said particular exemption mechanism
8 to be implemented; and
9 in response to a determination that said exempt encryption limitations allow said
10 particular exemption mechanism to be implemented, deriving said restrictions from said
11 set of exempt encryption limitations.

1 24. (Not Amended) In a system comprising an application and an
2 implementation class which provides an implementation for a particular service, a
3 framework comprising:
4 a mechanism for receiving a request from an application for a customized
5 implementation of a particular service;
6 a mechanism for instantiating an implementation class which provides an
7 implementation for the particular service to give rise to an implementation instance;
8 a mechanism for determining a set of zero or more restrictions to be imposed on
9 said customized implementation;
10 a mechanism for instantiating a wrapper class to give rise to a wrapper instance,
11 said wrapper instance comprising enforcement logic for enforcing said restrictions;

12 a mechanism for encapsulating said implementation instance and said restrictions
13 within said wrapper instance; and
14 a mechanism for providing said wrapper instance to the application as said
15 customized implementation.

1 25. (Not Amended) The framework of claim 24, wherein said wrapper
2 instance is invocable by the application without further interaction with the framework.

1 26. (Not Amended) The framework of claim 24, wherein the implementation
2 class provides an unrestricted implementation for the particular service.

1 27. (Not Amended) The framework of claim 26, wherein the particular service
2 is an encryption/decryption service, and wherein the unrestricted implementation
3 provided by the implementation class is capable of using unlimited encryption/decryption
4 parameters.

1 28. (Not Amended) The framework of claim 27, wherein the unrestricted
2 implementation provided by the implementation class is capable of using
3 encryption/decryption keys of any size.

1 29. (Not Amended) The framework of claim 24, wherein said enforcement
2 logic enforces said restrictions on said implementation instance.

1 30. (Not Amended) The framework of claim 29, wherein said enforcement
2 logic enforces said restrictions on said implementation instance by:
3 receiving a set of desired parameters from the application;
4 determining whether the desired parameters exceed said restrictions; and
5 in response to a determination that the desired parameters exceed said restrictions,
6 preventing said implementation instance from operating.

1 31. (Not Amended) The framework of claim 30, wherein said enforcement
2 logic is invoked upon initialization of said wrapper instance.

1 32. (Not Amended) The framework of claim 24, wherein the system further
2 comprises an exemption mechanism class which provides an implementation for a
3 particular exemption mechanism, and wherein said framework further comprises:
4 a mechanism for instantiating the exemption mechanism class to give rise to an
5 exemption mechanism instance; and
6 a mechanism for encapsulating said exemption mechanism instance within said
7 wrapper instance.

1 33. (Not Amended) The framework of claim 32, wherein said enforcement
2 logic is invoked upon initialization of said wrapper instance, and when invoked, said
3 enforcement logic:
4 determines whether said exemption mechanism instance has been invoked; and
5 in response to a determination that said exemption mechanism instance has not
6 been invoked, preventing said implementation instance from operating.

1 34. (Not Amended) The framework of claim 24, wherein said wrapper
2 instance comprises one or more invocable methods, wherein said implementation
3 instance comprises one or more invocable methods, and wherein the mechanism for
4 encapsulating comprises:
5 a mechanism for mapping the one or more invocable methods of said wrapper
6 instance to the one or more invocable methods of said implementation instance.

1 35. (Not Amended) The framework of claim 24, wherein the mechanism for
2 instantiating the implementation class comprises:
3 a mechanism for determining whether the implementation class is authentic; and
4 a mechanism for instantiating, in response to a determination that the
5 implementation class is authentic, the implementation class to give rise to said
6 implementation instance.

1 36. (Not Amended) The framework of claim 35, wherein the implementation
2 class has a digital signature associated therewith, and wherein the mechanism for
3 determining whether the implementation class is authentic comprises:
4 a mechanism for verifying said digital signature.

1 37. (Not Amended) The framework of claim 35, wherein the implementation
2 class authenticates the framework prior to giving rise to said implementation instance.

1 38. (Not Amended) The framework of claim 24, wherein the mechanism for
2 determining the set of zero or more restrictions comprises:

3 a mechanism for accessing information specifying one or more limitations; and
4 a mechanism for processing said limitations to derive said restrictions.

1 39. (Not Amended) The framework of claim 38, wherein the particular service
2 is an encryption/decryption service, and wherein said information comprises a set of one
3 or more default encryption limitations.

1 40. (Not Amended) The framework of claim 39, wherein said default
2 encryption limitations are derived by merging multiple jurisdiction policies and
3 extracting therefrom the most restrictive encryption limitations.

1 41. (Not Amended) The framework of claim 24, wherein the mechanism for
2 determining the set of zero or more restrictions comprises:
3 a mechanism for accessing information specifying one or more limitations;
4 a mechanism for determining permissions, if any, granted to the application; and
5 a mechanism for reconciling said limitations and said permissions to derive said
6 restrictions.

1 42. (Not Amended) The framework of claim 41, wherein said limitations and
2 said permissions are reconciled to derive restrictions which are least restrictive.

1 43. (Not Amended) The framework of claim 41, wherein the particular service
2 is an encryption/decryption service, and wherein said information comprises a set of one

3 or more default encryption limitations, and a set of zero or more exempt encryption
4 limitations which apply when one or more exemption mechanisms are implemented.

1 44. (Not Amended) The framework of claim 43, wherein said default
2 encryption limitations and said exempt encryption limitations are derived by merging
3 multiple jurisdiction policies and extracting therefrom the most restrictive encryption
4 limitations.

1 45. (Not Amended) The framework of claim 43, wherein the mechanism for
2 reconciling said limitations and said permissions comprises:

3 a mechanism for determining whether the application has been granted any
4 permissions; and

5 a mechanism for deriving, in response to a determination that the application has
6 not been granted any permissions, said restrictions from said set of default encryption
7 limitations.

1 46. (Not Amended) The framework of claim 43, wherein the mechanism for
2 reconciling said limitations and said permissions comprises:

3 a mechanism for determining whether the application has been granted any
4 permissions which require implementation of a particular exemption mechanism;

5 a mechanism for determining, in response to a determination that the application
6 has been granted a permission which requires implementation of a particular exemption
7 mechanism, whether said exempt encryption limitations allow said particular exemption
8 mechanism to be implemented; and

9 a mechanism for deriving, in response to a determination that said exempt
10 encryption limitations allow said particular exemption mechanism to be implemented,
11 said restrictions from said set of exempt encryption limitations.

1 47. (Not Amended) In a system comprising an application and an
2 implementation class which provides an implementation for a particular service, a
3 computer readable medium having stored thereon instructions which, when executed by
4 one or more processors, cause the one or more processors to implement a framework
5 which dynamically constructs a customized implementation, said computer readable
6 medium comprising:
7 instructions for causing one or more processors to receive a request from an
8 application for a customized implementation of a particular service;
9 instructions for causing one or more processors to instantiate an implementation
10 class which provides an implementation for the particular service to give rise to an
11 implementation instance;
12 instructions for causing one or more processors to determine a set of zero or more
13 restrictions to be imposed on said customized implementation;
14 instructions for causing one or more processors to instantiate a wrapper class to
15 give rise to a wrapper instance, said wrapper instance comprising enforcement logic for
16 enforcing said restrictions;
17 instructions for causing one or more processors to encapsulate said
18 implementation instance and said restrictions within said wrapper instance; and
19 instructions for causing one or more processors to provide said wrapper instance
20 to the application as said customized implementation.

1 48. (Not Amended) The computer readable medium of claim 47, wherein said
2 wrapper instance is invocable by the application without further interaction with the
3 framework.

1 49. (Not Amended) The computer readable medium of claim 47, wherein the
2 implementation class provides an unrestricted implementation for the particular service.

1 50. (Not Amended) The computer readable medium of claim 49, wherein the
2 particular service is an encryption/decryption service, and wherein the unrestricted
3 implementation provided by the implementation class is capable of using unlimited
4 encryption/decryption parameters.

1 51. (Not Amended) The computer readable medium of claim 50, wherein the
2 unrestricted implementation provided by the implementation class is capable of using
3 encryption/decryption keys of any size.

1 52. (Not Amended) The computer readable medium of claim 47, wherein said
2 enforcement logic enforces said restrictions on said implementation instance.

1 53. (Not Amended) The computer readable medium of claim 52, wherein said
2 enforcement logic enforces said restrictions on said implementation instance by:
3 receiving a set of desired parameters from the application;
4 determining whether the desired parameters exceed said restrictions; and

5 in response to a determination that the desired parameters exceed said restrictions,
6 preventing said implementation instance from operating.

1 54. (Not Amended) The computer readable medium of claim 53, wherein said
2 enforcement logic is invoked upon initialization of said wrapper instance.

1 55. (Not Amended) The computer readable medium of claim 47, wherein the
2 system further comprises an exemption mechanism class which provides an
3 implementation for a particular exemption mechanism, and wherein said computer
4 readable medium further comprises:

5 instructions for causing one or more processors to instantiate the exemption
6 mechanism class to give rise to an exemption mechanism instance; and

7 instructions for causing one or more processors to encapsulate said exemption
8 mechanism instance within said wrapper instance.

1 56. (Not Amended) The computer readable medium of claim 55, wherein said
2 enforcement logic is invoked upon initialization of said wrapper instance, and when
3 invoked, said enforcement logic:

4 determines whether said exemption mechanism instance has been invoked; and

5 in response to a determination that said exemption mechanism instance has not
6 been invoked, preventing said implementation instance from operating.

1 57. (Not Amended) The computer readable medium of claim 47, wherein said
2 wrapper instance comprises one or more invocable methods, wherein said

3 implementation instance comprises one or more invocable methods, and wherein the
4 instructions for causing one or more processors to encapsulate comprises:
5 instructions for causing one or more processors to map the one or more invocable
6 methods of said wrapper instance to the one or more invocable methods of said
7 implementation instance.

1 58. (Not Amended) The computer readable medium of claim 47, wherein the
2 instructions for causing one or more processors to instantiate the implementation class
3 comprises:
4 instructions for causing one or more processors to determine whether the
5 implementation class is authentic; and
6 instructions for causing one or more processors to instantiate, in response to a
7 determination that the implementation class is authentic, the implementation class to give
8 rise to said implementation instance.

1 59. (Not Amended) The computer readable medium of claim 58, wherein the
2 implementation class has a digital signature associated therewith, and wherein the
3 instructions for causing one or more processors to determine whether the implementation
4 class is authentic comprises:
5 instructions for causing one or more processors to verify said digital signature.

1 60. (Not Amended) The computer readable medium of claim 58, wherein the
2 implementation class authenticates the framework prior to giving rise to said
3 implementation instance.

1 61. (Not Amended) The computer readable medium of claim 47, wherein the
2 instructions for causing one or more processors to determine the set of zero or more
3 restrictions comprises:

4 instructions for causing one or more processors to access information specifying
5 one or more limitations; and

6 instructions for causing one or more processors to process said limitations to
7 derive said restrictions.

1 62. (Not Amended) The computer readable medium of claim 61, wherein the
2 particular service is an encryption/decryption service, and wherein said information
3 comprises a set of one or more default encryption limitations.

1 63. (Not Amended) The computer readable medium of claim 62, wherein said
2 default encryption limitations are derived by merging multiple jurisdiction policies and
3 extracting therefrom the most restrictive encryption limitations.

1 64. (Not Amended) The computer readable medium of claim 47, wherein the
2 instructions for causing one or more processors to determine the set of zero or more
3 restrictions comprises:

4 instructions for causing one or more processors to access information specifying
5 one or more limitations;

6 instructions for causing one or more processors to determine permissions, if any,
7 granted to the application; and

8 instructions for causing one or more processors to reconcile said limitations and
9 said permissions to derive said restrictions.

1 65. (Not Amended) The computer readable medium of claim 64, wherein said
2 limitations and said permissions are reconciled to derive restrictions which are least
3 restrictive.

1 66. (Not Amended) The computer readable medium of claim 64, wherein the
2 particular service is an encryption/decryption service, and wherein said information
3 comprises a set of one or more default encryption limitations, and a set of zero or more
4 exempt encryption limitations which apply when one or more exemption mechanisms are
5 implemented.

1 67. (Not Amended) The computer readable medium of claim 66, wherein said
2 default encryption limitations and said exempt encryption limitations are derived by
3 merging multiple jurisdiction policies and extracting therefrom the most restrictive
4 encryption limitations.

1 68. (Not Amended) The computer readable medium of claim 66, wherein the
2 instructions for causing one or more processors to reconcile said limitations and said
3 permissions comprises:

4 instructions for causing one or more processors to determine whether the
5 application has been granted any permissions; and

6 instructions for causing one or more processors to derive, in response to a
7 determination that the application has not been granted any permissions, said restrictions
8 from said set of default encryption limitations.

1 69. (Not Amended) The computer readable medium of claim 66, wherein the
2 instructions for causing one or more processors to reconcile said limitations and said
3 permissions comprises:

4 instructions for causing one or more processors to determine whether the
5 application has been granted any permissions which require implementation of a
6 particular exemption mechanism;

7 instructions for causing one or more processors to determine, in response to a
8 determination that the application has been granted a permission which requires
9 implementation of a particular exemption mechanism, whether said exempt encryption
10 limitations allow said particular exemption mechanism to be implemented; and

11 instructions for causing one or more processors to derive, in response to a
12 determination that said exempt encryption limitations allow said particular exemption
13 mechanism to be implemented, said restrictions from said set of exempt encryption
14 limitations.

1 70. (New) The method of claim 1, wherein determining said set of zero or
2 more restrictions includes determining a set of zero or more restrictions that are specific
3 to said application.

1 71. (New) The method of claim 70, wherein determining said set of zero or
2 more restrictions that are specific to said application includes determining a set of zero or
3 more restrictions that are customized for said application.

1 72. (New) The method of claim 1, wherein said set is a first set, and wherein
2 said customized implementation is a first customized implementation, and further
3 comprising:

4 receiving a request from a second application for a second customized
5 implementation of said particular service, wherein said second customized
6 implementation differs from said first customized implementation;

7 instantiating said implementation class which provides said implementation for
8 said particular service to give rise to a second implementation instance;

9 determining a second set of zero or more restrictions to be imposed on said
10 second customized implementation, wherein said second set differs from said first set;

11 instantiating said wrapper class to give rise to a second wrapper instance, said
12 second wrapper instance comprising enforcement logic for enforcing said second set of
13 zero or more restrictions;

14 encapsulating said second implementation instance and said second set of zero or
15 more restrictions within said second wrapper instance; and

16 providing said second wrapper instance to said second application as said second
17 customized implementation.

1 73. (New) The framework of claim 24, wherein said mechanism for
2 determining said set of zero or more restrictions includes a mechanism for determining a
3 set of zero or more restrictions that are specific to said application.

1 74. (New) The framework of claim 73, wherein said mechanism for
2 determining said set of zero or more restrictions that are specific to said application
3 includes a mechanism for determining a set of zero or more restrictions that are
4 customized for said application.

1 75. (New) The framework of claim 24, wherein said set is a first set, and
2 wherein said customized implementation is a first customized implementation, and
3 further comprising:

4 a mechanism for receiving a request from a second application for a second
5 customized implementation of said particular service, wherein said second customized
6 implementation differs from said first customized implementation;

7 a mechanism for instantiating said implementation class which provides said
8 implementation for said particular service to give rise to a second implementation
9 instance;

10 a mechanism for determining a second set of zero or more restrictions to be
11 imposed on said second customized implementation, wherein said second set differs from
12 said first set;

13 a mechanism for instantiating said wrapper class to give rise to a second wrapper
14 instance, said second wrapper instance comprising enforcement logic for enforcing said
15 second set of zero or more restrictions;

16 a mechanism for encapsulating said second implementation instance and said
17 second set of zero or more restrictions within said second wrapper instance; and
18 a mechanism for providing said second wrapper instance to said second
19 application as said second customized implementation.

1 76. (New) The computer readable medium of claim 49, wherein said
2 instructions for determining said set of zero or more restrictions include instructions for
3 determining a set of zero or more restrictions that are specific to said application.

1 77. (New) The computer readable medium of claim 76, wherein said
2 instructions for determining said set of zero or more restrictions that are specific to said
3 application include instructions for determining a set of zero or more restrictions that are
4 customized for said application.

1 78. (New) The computer readable medium of claim 49, wherein said set is a
2 first set, and wherein said customized implementation is a first customized
3 implementation, and further comprising:
4 instructions for receiving a request from a second application for a second
5 customized implementation of said particular service, wherein said second customized
6 implementation differs from said first customized implementation;
7 instructions for instantiating said implementation class which provides said
8 implementation for said particular service to give rise to a second implementation
9 instance;

10 instructions for determining a second set of zero or more restrictions to be
11 imposed on said second customized implementation, wherein said second set differs from
12 said first set;
13 instructions for instantiating said wrapper class to give rise to a second wrapper
14 instance, said second wrapper instance comprising enforcement logic for enforcing said
15 second set of zero or more restrictions;
16 instructions for encapsulating said second implementation instance and said
17 second set of zero or more restrictions within said second wrapper instance; and
18 instructions for providing said second wrapper instance to said second application
19 as said second customized implementation.